Bayesian Hierarchical MPT Models

Applications with TreeBUGS

Daniel W. Heck

Philipps Universität Marburg

25.02.2020

TreeBUGS: Bayesian Hierarchical MPT Modeling

1) The R package `TreeBUGS`
2) Basic modeling
   - Model fitting
   - Convergence
   - Plots
   - Model fit

Software for Hierarchical MPT Models

Software for Hierarchical MPTs

- Implementation of MCMC sampling in R/C/Fortran (Klauer 2010)
- General-purpose software: WinBUGS/JAGS/Stan (Matzke et al. 2015)
- Requires re-implementation of summaries, statistics, plots

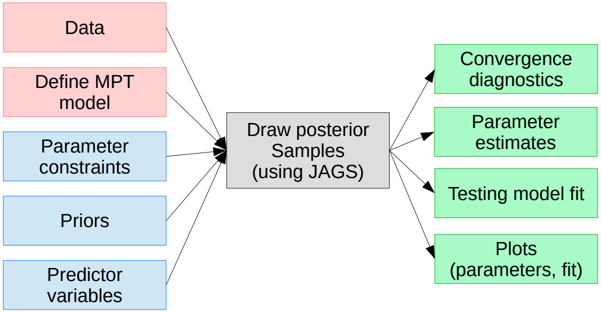
TreeBUGS: A user-friendly R package (Heck, Arnold, and Arnold 2018)

- Easy-to-use, open source, free
- Fitting and testing MPT models
  - Posterior sampling, summary statistics, and plots
  - Data generation, robustness simulations
  - Change priors, add predictors, etc.
  - Current limitation: Crossed random effects (for persons & items)

TreeBUGS

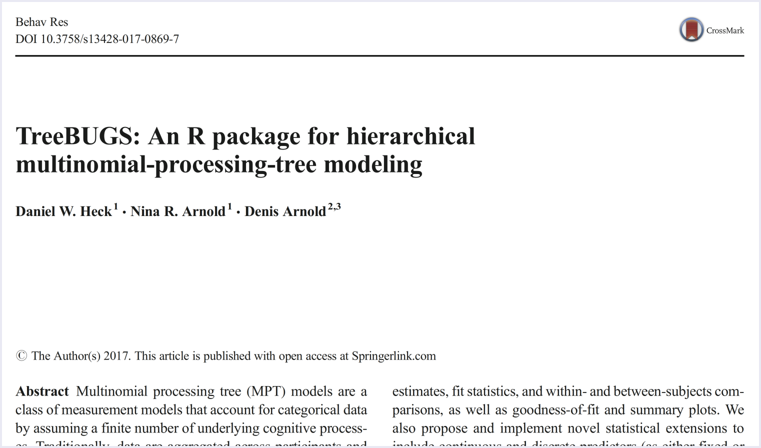
Functionality of TreeBUGS

- Input: R objects or text/csv files (minimal R knowledge required)
- Priors and other details can be changed in R
- TreeBUGS translates the model to JAGS (Plummer, 2003) to draw posterior samples
- Functions for post-processing, summaries and plots



Reference

TreeBUGS Paper



Behav Res
DOI 10.3758/s13428-017-0869-7

CrossMark

TreeBUGS: An R package for hierarchical multinomial-processing-tree modeling

Daniel W. Heck[1] · **Nina R. Arnold**[1] · **Denis Arnold**[2,3]

Abstract Multinomial processing tree (MPT) models are a class of measurement models that account for categorical data by assuming a finite number of underlying cognitive process-es. Traditionally, data are aggregated across participants and

estimates, fit statistics, and within- and between-subjects com-parisons, as well as goodness-of-fit and summary plots. We also propose and implement novel statistical extensions to include continuous and discrete predictors (as either fixed or

Basic Modeling

(corresponding R script: `04-application-TreeBUGS.R`)

Overview

Modeling with TreeBUGS is simple

1. Specify model and data
2. Draw MCMC samples
3. Check convergence
4. Check model fit
5. Interpret/plot parameters

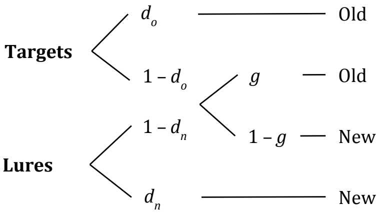
Note that these are the usual steps in any Bayesian analysis. . .

MPT Model Specification

- MPT structure is defined in an EQN model file
  - Can be copied from multiTree
- Difference: the symbol # allows to add comments

Two-high threshold model
(file: `2htm.eqn`)

```
# Targets
target  hit  do
target  hit  (1-do)*g
target  miss (1-do)*  (1-g)

# Lures
lure  cr  dn
lure  fa  (1-dn)*g
lure  cr  (1-dn)*(1-g)
```



Data Structure

- Data: Response frequencies in wide format
    - One line per person
    - One category per column
    - Column names must be identical to the EQN categories!
- Either supplied in .csv-file or as `data.frame` / `matrix` in R

Example: `2htm.csv`

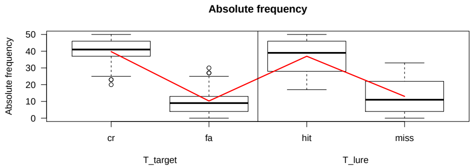
```r
# set working directory:
# setwd("D:/R/MPT-workshop/")
frequencies <- read.csv("2htm.csv")
head(frequencies, 5)
```

```
##   cr fa hit miss
## 1 47  3  45    5
## 2 38 12  22   28
## 3 37 13  21   29
## 4 37 13  32   18
## 5 46  4  44    6
```

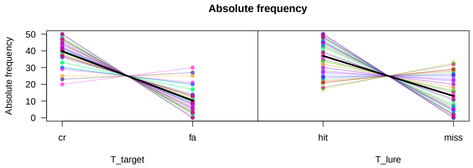
Heterogeneity

- Load TreeBUGS and plot heterogeneity
- If response frequencies are homogeneous, standard (fixed-effects) MPT models are statistically more efficient

```r
library(TreeBUGS)
plotFreq(frequencies, eqnfile = "2htm.eqn")
```



```r
plotFreq(frequencies, boxplot = FALSE, eqnfile = "2htm.eqn")
```



Fitting MPT Models

- Fitting an MPT model in TreeBUGS
  - Model: Text file in EQN syntax (with model equations)
  - Data: .csvfile
  - Constraints: text file with equality contraints

```r
fit <- traitMPT(eqnfile = "htm.txt",
                data = "responses.csv",
                restrictions = "2htm_constraints.txt")

# beta-MPT: different function, but identical arguments
fit_beta <- betaMPT(eqnfile = "htm.txt",
                    data = "responses.csv",
                    restrictions = "2htm_constraints.txt")
```

Fitting an MPT Model in R

- Alternative: define everything directly in R
  - Model: Text string (`character` in apostrophes)
  - Data: Matrix or data frame
  - Constraints: A list

```
htm <- "
target  hit  do
target  hit  (1-do)*g
target  miss (1-do)*  (1-g)

lure  cr  dn
lure  fa  (1-dn)*g
lure  cr  (1-dn)*(1-g)
"
fit <- traitMPT(eqnfile = htm,
                data = frequencies,
                restrictions = list("dn=do", "g=.50"))
```

Parameter Constraints

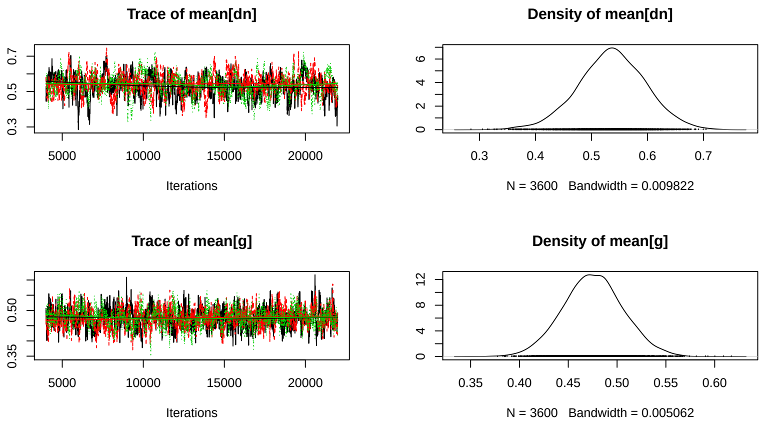
Equality constraints

```
# (A) use a general model file and constrain parameters:
fit <- traitMPT(eqnfile = htm,
                data = frequencies,
                restrictions = list("dn=do", "g=.50"))

# (B) hard-coding of constraints in the EQN file:
htm_constr <- "
target  hit  d
target  hit  (1-d)*.50
target  miss (1-d)*.50

lure  cr  d
lure  fa  (1-d)*.50
lure  cr  (1-d)*.50
"
fit <- traitMPT(eqnfile = htm_constr,
                data = frequencies)
```

Convergence

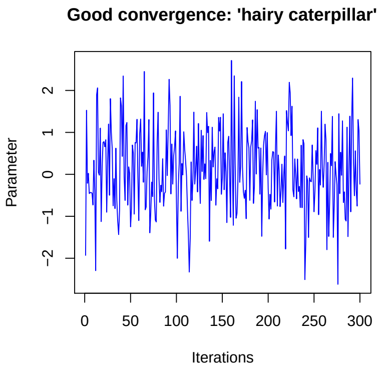
- Convergence check
  - Posterior/ MCMC samples should look unsystematic (like a hairy caterpillar)
  - For more options, see: `?plot.traitMPT`

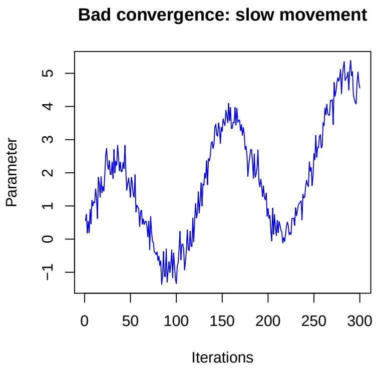
```r
plot(fit, parameter = "mean", type = "default")
```



Convergence

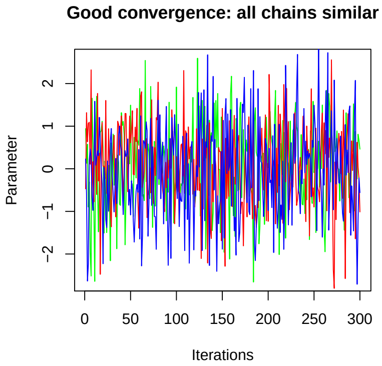
- Interpreting MCMC plots

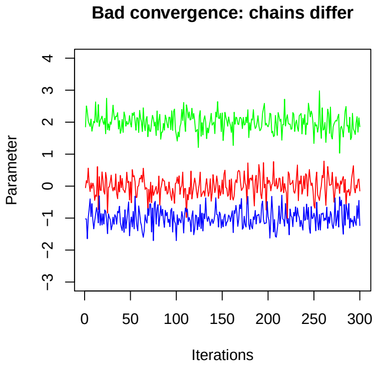




Convergence

- Gelman-Rubin statistic ($\hat{R}$)
  - Also known as: "potential scale reduction factor" or "R hat"
  - Similar to ANOVA: Compares between-chain and within-chain variances (large differences between these variances indicate nonconvergence)
  - Statistic should be close to 1 (standard criterion: $\hat{R} < 1.05$)





Convergence

- Gelman-Rubin statistic ($\hat{R}$)
  - Columns `Rhat` and `R_95%` in the summary output

```
summary(fit)
```

```
## Call:
## traitMPT(eqnfile = htm, data = frequencies, restrictions = list("dn=do"),
##     ppp = 1000)
##
## Group-level medians of MPT parameters (probability scale):
##          Mean    SD  2.5%   50% 97.5% Time-series SE n.eff  Rhat R_95%
## mean_dn 0.536 0.060 0.414 0.537 0.654          0.003   354 1.027 1.094
## mean_g  0.475 0.031 0.415 0.475 0.537          0.001   685 1.005 1.016
##
## Mean/Median of latent-trait values (probit-scale) across individuals:
##                 Mean    SD   2.5%    50% 97.5% Time-series SE n.eff  Rhat R_95%
## latent_mu_dn   0.092 0.154 -0.216  0.093 0.396          0.008   354 1.027 1.095
## latent_mu_g   -0.062 0.078 -0.215 -0.062 0.093          0.003   684 1.005 1.016
##
## Standard deviation of latent-trait values (probit scale) across individuals:
##                   Mean    SD  2.5%   50% 97.5% Time-series SE n.eff  Rhat R_95%
## latent_sigma_dn 1.062 0.137 0.828 1.052 1.368          0.002  3439 1.005 1.019
## latent_sigma_g  0.433 0.070 0.311 0.428 0.585          0.001  2890 1.005 1.018
##
## Correlations of latent-trait values on probit scale:
##              Mean   SD   2.5%   50% 97.5% Time-series SE n.eff  Rhat R_95%
## rho[dn,g] 0.315 0.17 -0.041 0.323 0.617          0.003  2380 1.002 1.005
##
## Correlations (posterior mean estimates) in matrix form:
```

Options for MCMC Sampling

- If the model has not converged, it must be fitted with more conservative settings:

```
fit <- traitMPT(
  eqnfile = htm, data = frequencies,
  restrictions = list("dn=do"),

  n.adapt = 5000, # longer adaption of JAGS increases efficiency of sampling
  n.burnin = 5000,# longer burnin avoids issues due to bad starting values
  n.iter = 30000, # drawing more MCMC samples leads to higher precision
  n.thin = 10,    # ommitting every 10th sample reduces memory load
  n.chains = 4)   # more MCMC chains increase precision
```

```
## MCMC sampling started at  2020-02-20 12:38:26
## Calling 4 simulations using the parallel method...
## Following the progress of chain 1 (the program will wait for all chains
## to finish before continuing):
## Welcome to JAGS 4.3.0 on Thu Feb 20 12:38:29 2020
## JAGS is free software and comes with ABSOLUTELY NO WARRANTY
## Loading module: basemod: ok
## Loading module: bugs: ok
## . Loading module: dic: ok
## . Loading module: glm: ok
## . . Reading data file data.txt
## . Compiling data graph
##    Resolving undeclared variables
```

Extend MCMC Sampling

- If the MCMC samples are OK but higher precision is needed:
  - Extend sampling and add new MCMC samples to the fitted JAGS object

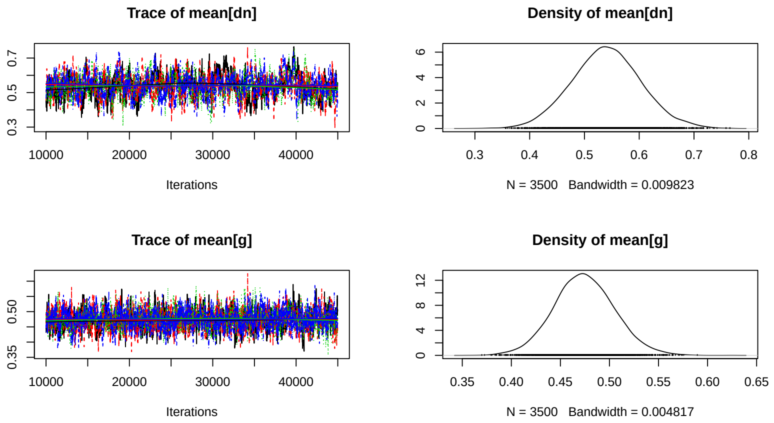
```
fit2 <- extendMPT(fit,                # fitted MPT model
                  n.adapt = 2000,     # JAGS need to restart and adapt again
                  n.burnin = 0,       # burnin not needed if previous samples are OK
                  n.iter = 10000)     # how many additional iterations?
```

```
## Calling 4 simulations using the parallel method...
## Following the progress of chain 1 (the program will wait for all chains
## to finish before continuing):
## Welcome to JAGS 4.3.0 on Thu Feb 20 12:38:58 2020
## JAGS is free software and comes with ABSOLUTELY NO WARRANTY
## Loading module: basemod: ok
## Loading module: bugs: ok
## . Loading module: dic: ok
## . Loading module: glm: ok
## . . Reading data file data.txt
## . Compiling data graph
##    Resolving undeclared variables
##    Allocating nodes
##    Initializing
##    Reading data back into data table
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 100
##    Unobserved stochastic nodes: 55
```

Convergence for Second Fit

- Check convergence again:

```r
plot(fit2, parameter = "mean", type = "default")
```



Parameter Estimates

- Summary statistics for posterior distribution:
  - Posterior mean and median (50% quantile)
  - Posterior standard deviation (SD, similar to standard error)
  - Bayesian credibility interval (2.5% and 97.5% quantiles)

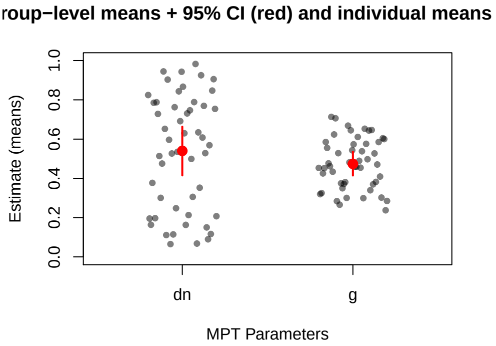
```r
summary(fit2)
```

```
## Call:
## [[1]]
## traitMPT(eqnfile = htm, data = frequencies, restrictions = list("dn=do"),
##     n.iter = 30000, n.adapt = 5000, n.burnin = 5000, n.thin = 10,
##     n.chains = 4)
##
## [[2]]
## extendMPT(fittedModel = fit, n.iter = 10000, n.adapt = 2000,
##     n.burnin = 0)
##
##
## Group-level medians of MPT parameters (probability scale):
##          Mean    SD  2.5%   50% 97.5% Time-series SE n.eff  Rhat R_95%
## mean_dn 0.540 0.063 0.417 0.540 0.668          0.002   698 1.006 1.018
## mean_g  0.474 0.031 0.415 0.474 0.537          0.001  1595 1.002 1.005
##
## Mean/Median of latent-trait values (probit-scale) across individuals:
##                 Mean    SD   2.5%    50% 97.5% Time-series SE n.eff  Rhat R_95%
## latent_mu_dn   0.102 0.162 -0.210  0.100 0.435          0.006   696 1.006 1.018
## latent_mu_g   -0.064 0.078 -0.216 -0.066 0.094          0.002  1594 1.002 1.005
##
## Standard deviation of latent-trait values (probit scale) across individuals:
##                   Mean   SD  2.5%   50% 97.5% Time-series SE n.eff  Rhat R_95%
## latent_sigma_dn 1.066 0.14 0.828 1.054 1.376          0.002  4662 1.004 1.011
## latent_sigma_g  0.432 0.07 0.311 0.428 0.586          0.001  6041 1.000 1.001
```

Plot Parameter Estimates

- Estimates for group-level parameters
  - Overall mean μ: Posterior mean and Bayesian credibility interval
  - Individual parameters θ_i: Posterior mean

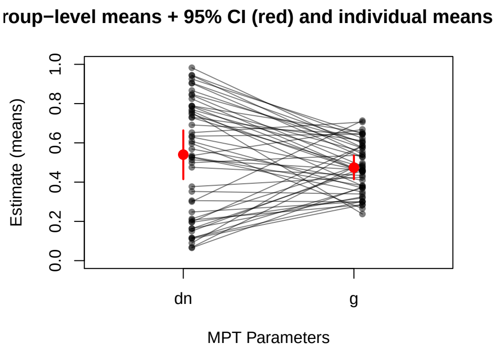
```
plotParam(fit)
```



Plot Parameter Estimates

- Plot parameter profiles per person
- For instance, to assess the test-retest reliability of a parameter (Michalkiewicz & Erdfelder, 2016)
- For more options, see: `?plotParam`

```r
plotParam(fit, addLines = TRUE, select = c("dn", "g"))
```



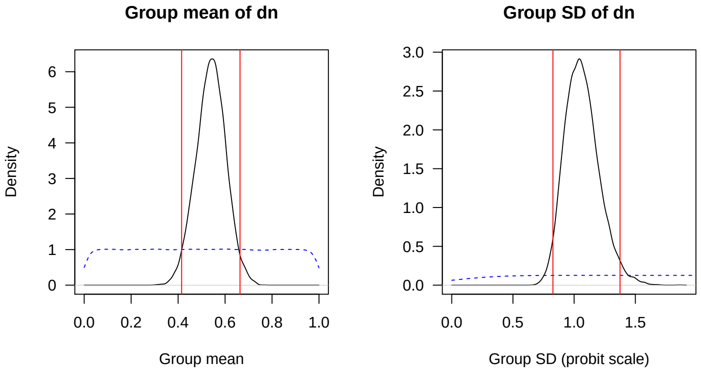
Compare Prior and Posterior

How much did we learn about the parameters?

- Graphical assessment: Plot prior (blue) and posterior (black) densities

```
plotPriorPost(fit)
```

```
## Press <Enter> to show the next plot.
```



```
## Press <Enter> to show the next plot.
```

Model Fit: Predicted vs. Observed Data

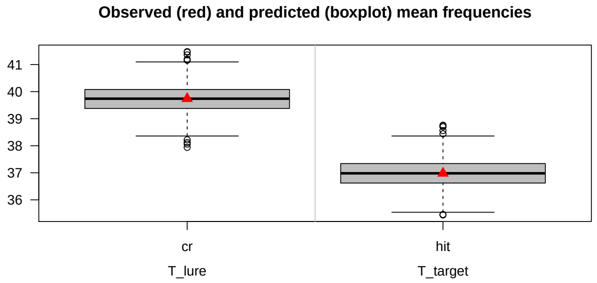
Graphical test of model fit

- Plot means of observed frequencies
- Compare against posterior-predicted frequencies (boxplots)

```
colMeans(frequencies)  # observed group means that are tested
```

```
##    cr    fa   hit  miss
## 39.74 10.26 36.98 13.02
```

```
plotFit(fit)             # graphical test
```



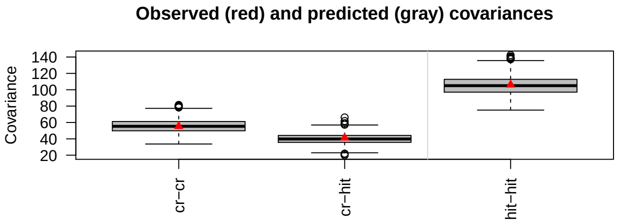
Model Fit: Predicted vs. Observed Data

- Model fit for *covariances* of the observed frequencies

```
cov(frequencies)            # observed covariance matrix that is tested
```

```
##             cr        fa        hit       miss
## cr    55.38000 -55.38000   41.21918  -41.21918
## fa   -55.38000  55.38000  -41.21918   41.21918
## hit   41.21918 -41.21918  106.02000 -106.02000
## miss -41.21918  41.21918 -106.02000  106.02000
```

```
plotFit(fit, stat = "cov")  # graphical test
```



Model Fit: Test Statistic for Predicted vs. Observed Data

Testing model fit

- We need test statistics to quantify model fit (Klauer, 2010)
  - T1 statistic: Mean structure of frequences
  - T2 statistic: Covariance matrix of frequencies
  - Similar to Pearson's X^2: discrepancy between observed and expected data
- Posterior predictive p-value (PPP) measures model fit:
  1) Compute T1 for the observed data
  2) Compute T1 for the posterior predicted data
  3) PPP $=$ probability that T1(predicted) is larger than T1(observed)
- Interpretation
  - Values around .50 indicate good model fit
  - Values close to 0 indicate misfit
  - In contrast to frequentist p-values, PPP values are *not* uniformly distributed when generating data from the correct model

```
PPP(fit, M = 2000, nCPU = 4)
```

```
##  ## Mean structure (T1):
##  Observed =  0.0325821 ; Predicted =  0.03440444 ; p-value =  0.513
##
## ## Covariance structure (T2):
##  Observed =  7.316926 ; Predicted =  7.525546 ; p-value =  0.51
##
## ## Individual fit (T1):
##     1     2     3     4     5     6     7     8     9    10    11    12    13
## 0.530 0.480 0.508 0.526 0.542 0.496 0.534 0.518 0.504 0.417 0.538 0.335 0.378
##    14    15    16    17    18    19    20    21    22    23    24    25    26
```

Appendix

Appendix: Testing for Heterogeneity

- Test by Smith and Batchelder (2008)
  - A) Test person heterogeneity assuming items homogeneity (χ^2)
  - B) Test person heterogeneity under item heterogeneity (permutation bootstrap)

```
# A) chi^2 test
test <- testHetChi(freq = frequencies,
                   tree = c(cr="lure", fa="lure",
                            hit="target",miss="target"))
data.frame(test)
```

```
##      chisq df          prob
## 1 872.2505 98 1.782941e-124
```

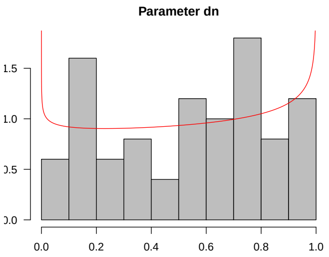
```
# B) requires data in long format (variables: person / item / response)
# testHetPerm(data, tree, source = "person")
```

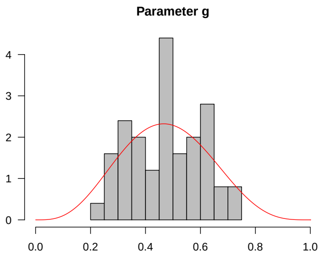
Appendix: Group-Level Distribution

Distribution of individual estimates

- Histogram: Distribution of θ estimates (posterior mean per person)
- Red density: Estimated group-level distribution

```
plotDistribution(fit)  # graphical test
```



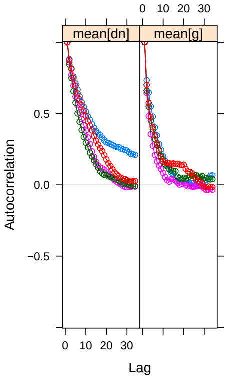


Appendix: Convergence

Autocorrelation function

- How strongly are the MCMC samples correlated between iteration t and iteration $t + \text{Lag}$?
- Ideally, these curves should rapidly decrease towards zero.

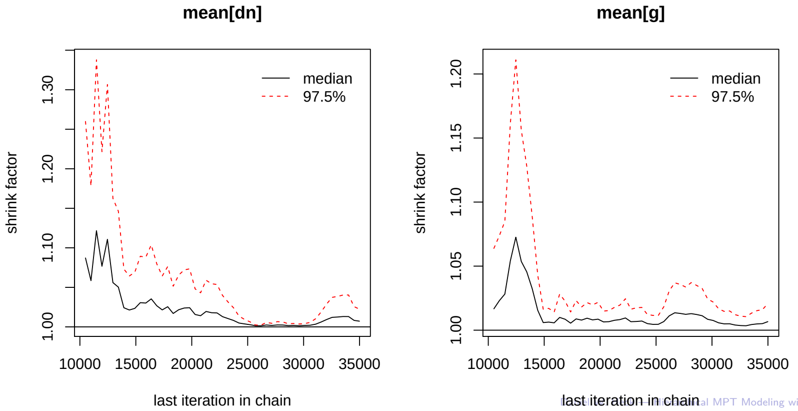
```r
plot(fit, parameter = "mean", type = "acf")
```



Appendix: Convergence

- Plot evolution of Gelman-Rubin statistic
  - Also known as: “potential scale reduction factor” or “R hat”
  - Similar to ANOVA: Compares between-chain and within-chain variances (large differences between these variances indicate nonconvergence)
  - Statistic should be close to 1

```r
plot(fit, parameter = "mean", type = "gelman")
```



Appendix: MPT Versions Implemented in TreeBUGS

MCMC samplers available in TreeBUGS

- Fixed-effects ("standard") MPT: Gibbs sampler in `C++`
- Beta-MPT: `JAGS` and `C++`
- Latent-trait with extensions: `JAGS`
    - Combination of random- and fixed-effects parameters
    - Continuous and categorical covariates
    - Group-level structure: Independent normal distributions

References I

Heck, Daniel W, Nina R. Arnold, and Denis Arnold. 2018. "TreeBUGS: An R Package for Hierarchical Multinomial-Processing-Tree Modeling." *Behavior Research Methods* 50: 264–84. https://doi.org/10.3758/s13428-017-0869-7.

Klauer, K. C. 2010. "Hierarchical Multinomial Processing Tree Models: A Latent-Trait Approach." *Psychometrika* 75: 70–98. https://doi.org/10.1007/s11336-009-9141-0.

Matzke, Dora, Conor V. Dolan, William H. Batchelder, and Eric-Jan Wagenmakers. 2015. "Bayesian Estimation of Multinomial Processing Tree Models with Heterogeneity in Participants and Items." *Psychometrika* 80: 205–35. https://doi.org/10.1007/s11336-013-9374-9.

Smith, J. B., and W. H. Batchelder. 2008. "Assessing Individual Differences in Categorical Data." *Psychonomic Bulletin & Review* 15: 713–31. https://doi.org/10.3758/PBR.15.4.713.